are several methods that are suitable for automatically determining offsets between loosely packed tiles. For example, as illustrated with respect to FIG. **46** (showing communicating tiles **652** and **654**, with respective displays **651** and **653**), optical encoding **660** along an edge can be used to identify tile orientation. It is possible to use a binary coded optical pattern along an edge which is regular and encodes the distance from a vertex at any point. A tile that abuts, or is relatively aligned, can read this pattern using optical sensors **658** and **659** and determine the display offset in the direction of the edge. Alternatively, as illustrated by FIG. **47** (showing communicating tiles **672** and **674**), it is possible to use radio based techniques that rely on signal strength triangulation. Each vertex **675** or **676** of a tile **672** can contain a wireless transmitter and a receiver. If these vertices transmit a short characteristic wireless signal at well known times, a nearby tile **674** can use the receivers situated at its own vertices **680**, **681**, **682**, and **683** to triangulate the position of each transmitting vertex relative to them by measuring the relative delay by which the signal was received. After two of the vertices of the original transmitting tile **672** have signaled, the adjoining tile **674** can determine its exact position and orientation within their local region of a tile array. The transmitting and receiving tiles can now swap roles, with the result that both tiles will know their relative position. This process can be extended across the tile array.

Free format tiling differs from loose-packed tiling in that there are no proximity, or regular format, constraints. To demonstrate how a free format tiling system might work, the following example is described. Imagine a number of laptop computers each fitted with a Global Positioning System (GPS) and a radio modem. Each laptop can determine its position (with acceptable error) and communicate it to all the other laptops by contacting them through the radio modem. After some period of time the laptops will all know their relative location and absolute location. If any computers changes its location, it can update the local neighbors to ensure there is enough understanding in the array to consider the computers are in a known tiling configuration even though they are not close to each other and in fact they may be in different geographic regions. An application that might use this free format tiling system is one that wishes to ensure information is being correctly sent in a uniform and dispersed fashion across a very large area. For example, the information that each laptop computer receives could be instructions to release an amount of a pesticide used to control an insect that does damage to commercial crops. If the pesticide is released in too high a concentration in any locality, it may be hazardous to human health. The tiled approach allows the computers to roam (e.g. in the back of a truck), displaying information as to the type and concentration of pesticide that is to be released given their current relative proximity.

Various conventional algorithms can be used to support distribution of information between autonomous tiled displays. These algorithms assume a system by which there is a master controller generating data to be displayed. There is also a large array of tiled computers that the system will use to display visual data and/or process information. Each computer in the tiled array contains a unique ID. It is the job of the master to split the data into pieces that each computer can display and for this information to be packaged along with the ID of the target computer. The algorithms below describe how the information travels from the master to a destination display tile:

Daisy Chain Routing

Display tiles are arranged to have a logical connectivity with each other so that each one only transmits information to the next in a predefined line. The computers are said to be daisy chained to each. Any information sent to the start of the chain contains an ID and the first computer in the chain compares it to its own ID. If it matches, it acts on the data. If not, it sends the data on to the next computer in the chain until it finds its destination.

N-ary Routing

In N-ary routing the path to the destination is contained in the ID of the device. A simple routing example is schematically illustrated with reference to the direction arrows **695** in FIG. **48**, which shows quaternary routing in physically connected tileable displays. In quaternary routing, an array is conceptually arranged as a quaternary tree with each node having an input and three outputs. In this system each pair of bits of the ID contains a routing command. A **0** indicates send the packet to the first output, a **1** indicates the second output, a **2** indicates the third output, and **3** indicates no further transmission. There is also a count that gets decremented by each node to tell successive nodes the bit number that is currently being considered and when the packet has reached its destination. In this way packets are forwarded from node to node with a simple choice at each stage until they reach their target display. As will be appreciated, it is possible to design N-ary systems with more than three outputs (a power of two is usually convenient for implementation (e.g. 4, 8, 16 . . .)).

Flooding

Flooding has no predefined routing structure. The computer that takes the first packet from the master checks to see if it has the correct ID. If not, the packet is sent out on all links to which it has not yet sent or received that packet. The result is a flood of copies of the packet across the array eventually reaching its destination. The packets must also have a maximum hop count to ensure they eventually are removed from the system. The disadvantage of this approach is that many more tiles are burdened with the processing of unwanted data than in the previous two schemes, which may impact the overall efficiency of the system.

Hot Potato

The Hot Potato algorithm is similar to the Flooding algorithm, except a retransmitted packet is sent out on only one output that is either chosen randomly or is the least busy. The process stops when the packet reaches the correct tile. The time that a packet takes to reach its destination is not deterministic.

As those skilled in the art will appreciate, other various modifications, extensions, and changes to the foregoing disclosed embodiments of the present invention are contemplated to be within the scope and spirit of the invention as defined in the following claims.

The claimed invention is:

1. A plurality of tileable devices for transferring data, comprising

a first device having a display, a processor, and a first communication module for transferring data;

a second device having a display, a processor and a second communication module for transferring data;

a third device having a display, a processor and a third communication module for transferring data;

wherein the first device is connected in substantially simultaneous communication with the second device and the third device to pass data based on spatial positions of the respective first device, second device, and third device, and